1. Klausur zu Programmierung WS 2000/2001

Prof. Gert Smolka

22. Dezember 2000

Name und Vorname

Matrikelnummer

Hinweise:

- Bitte alle Lösungen direkt auf den Aufgabenblättern notieren.
- Hilfsmittel sind nicht zugelassen.
- Für die Bearbeitung der Klausur stehen 150 Minuten zur Verfügung. Insgesamt sind 150 Punkte erreichbar. Die Punkteverteilung gibt Ihnen also einen Anhaltspunkt, wieviel Zeit sie für jede Aufgabe verwenden sollten.
- Zum Bestehen der Klausur genügt die Hälfte (75) der maximal erreichbaren Punkte.

1	2	3	4	5	6	7	8	9	10	11	12	$\sum$
8	11	14	14	8	15	8	12	16	12	12	20	150

Note:	

Verwenden Sie bitte nur die folgenden vordefinierten Prozeduren:

```
@ : 'a list * 'a list -> 'a list

map : ('a -> 'b) -> 'a list -> 'b list

rev : 'a list -> 'a list

foldl : ('a * 'b -> 'b) -> 'b -> 'a list -> 'b

foldr : ('a * 'b -> 'b) -> 'b -> 'a list -> 'b

Listsort.sort : ('a * 'a -> order) -> 'a list -> 'a list
```

Außerdem dürfen Sie die folgenden Typdeklarationen als gegeben annehmen:

```
datatype order = LESS | EQUAL | GREATER

datatype 'a tree =
    L of 'a
  | N of 'a * 'a tree * 'a tree

datatype 'a term = T of 'a * 'a term list
```

Aufgabe 1: Schnelles Reversieren (8) Schreiben Sie eine Prozedur

```
reverse : 'a list -> 'a list
```

die Listen *xs* mit der Laufzeit $\Theta(|xs|)$ reversiert.

Aufgabe 2: Count (7+4)

(a) Schreiben Sie mit `foldl` eine Prozedur

```
count : ''a -> ''a list -> int
```

die für einen Wert die Anzahl seiner Auftreten in einer Liste berechnet.

(b) Geben Sie die Laufzeit von `count` für Listen *xs* mit Θ an.

Aufgabe 3: Until (7+7)

(a) Schreiben Sie eine Prozedur

```
until : (int -> bool) -> int
```

die zu einer Prozedur p das kleinste $n \in \mathbb{N}$ berechnet, für das p den Wert `true` liefert. Verwenden Sie eine Hilfsprozedur.

(b) Geben Sie mithilfe von `until` eine Abstraktion an, die eine Prozedur `int -> int` liefert, die zu $x \in \mathbb{N}$ das größte $n \in \mathbb{N}$ mit $n^2 \leq x$ berechnet.

Aufgabe 4: Dezimaldarstellung (7+7) Die Dezimaldarstellung einer natürlichen Zahl ist die Liste ihrer Ziffern. Beispielsweise hat 7856 die Dezimaldarstellung [7, 8, 5, 6].

(a) Schreiben Sie eine Prozedur

```
dec : int -> int list
```

die die Dezimaldarstellung einer natürlichen Zahl berechnet. Verwenden Sie `div` und `mod`.

(b) Schreiben Sie mithilfe von `foldl` eine Prozedur

```
int : int list -> int
```

die zu einer Dezimaldarstellung die dargestellte natürliche Zahl berechnet.

Aufgabe 5: Schnelles Potenzieren (8) Schreiben Sie eine Prozedur

```
potenz : int * int -> int
```

die zu $(x, n) \in \mathbb{Z} \times \mathbb{N}$ die Potenz x^n mit der Laufzeit $\Theta(\log n)$ berechnet.

Aufgabe 6: Quicksort (7+8) Sie sollen eine Sortierprozedur schreiben, die ganzahlige Listen mit dem Quicksort-Algorithmus sortiert.

(a) Schreiben Sie eine Prozedur

```
partition : int -> int list -> int list * int list
```

die zu einer Zahl x und einer Liste xs zwei Listen us und vs wie folgt berechnet:

(i) $us@vs$ ist eine Permutation von xs.

(ii) Alle Elemente von us sind echt kleiner als x und alle Elemente von vs sind größer gleich als x.

(b) Schreiben Sie eine Prozedur

```
qsort : int list -> int list
```

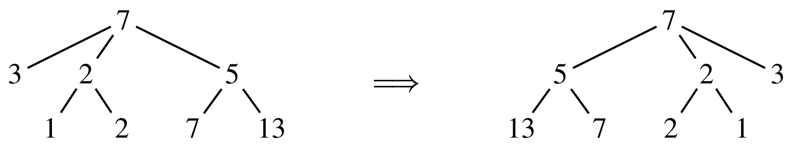
die Listen in absteigender (!) Ordnung sortiert.

Aufgabe 7: Map für Terme (8) Schreiben Sie eine Prozedur

```
mapterm : ('a -> 'b) -> 'a term -> 'b term
```

die eine Prozedur auf jede Marke eines Terms anwendet (analog zu `map` für Listen).

Aufgabe 8: Spiegeln von geordneten Bäumen (6+6) Spiegeln reversiert die Anordnung der Unterbäume eines geordneten Baums (auf allen Ebenen):



(a) Schreiben Sie eine Prozedur

```
mirror : 'a tree -> 'a tree
```

die Binärbäume spiegelt.

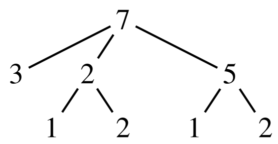
(b) Schreiben Sie eine Prozedur

```
mirror : 'a term  -> 'a term
```

die Terme spiegelt.

Aufgabe 9: Ebenen von geordneten Bäumen (8+8) Sei b ein geordneter Baum. Die *Tiefe* eines Knotens $u \in \mathit{Dom}\ b$ ist $|u|$ (also der Abstand zur Wurzel). Die *n-te Ebene von b* ($n \in \mathbb{N}$) ist die Liste der Marken aller Knoten der Tiefe n, angeordnet entsprechend der Anordnung der Knoten.

Als Beispiel betrachten wir den Baum



Die nullte Ebene ist [7], die erste Ebene ist [3, 2, 5], die zweite Ebene ist [1, 2, 1, 2], und die dritte und alle nachfolgenden Ebenen sind [].

(a) Schreiben Sie eine Prozedur

```
level : 'a tree * int -> 'a list
```

die für $n \in \mathbb{N}$ die n-te Ebene eines Binärbaums liefert.

(b) Schreiben Sie eine Prozedur

```
level : 'a term * int -> 'a list
```

die für $n \in \mathbb{N}$ die n-te Ebene eines Terms liefert.

Aufgabe 10: Schneller Test auf Doppelauftreten (12) Sie sollen eine Prozedur

```
test : int list -> bool
```

schreiben, die testet, ob in einer Liste ein Element mehrfach auftritt. Die Prozedur `Test` soll für alle $xs \in \mathcal{L}(\mathbb{Z})$ die Laufzeit $O(|xs| \log |xs|)$ haben. Verwenden Sie die Prozedur

```
Listsort.sort : ('a * 'a -> order) -> 'a list -> 'a list
```

die Listen xs mit der Laufzeit $\Theta(|xs| \log |xs|)$ sortiert.

Aufgabe 11: Listeninduktion (12) Sei X eine Menge und seien die folgenden rekursiven Funktionsdefinitionen gegeben:

$$@ \in \mathcal{L}(X) \times \mathcal{L}(X) \to \mathcal{L}(X)$$

$$\begin{aligned} nil@ys &= ys \\ (x :: xr)@ys &= x :: (xr@ys) \end{aligned}$$

$$|\_| \in \mathcal{L}(X) \to \mathbb{N}$$

$$\begin{aligned} |nil| &= 0 \\ |x :: xr| &= 1 + |xr| \end{aligned}$$

Beweisen Sie:

$$\forall xs \in \mathcal{L}(x)\ \forall ys \in \mathcal{L}(x): \ |xs@ys| = |xs| + |ys|$$

Aufgabe 12: Laufzeitsynthese (5+2+8+5) Schreiben Sie möglichst einfache Prozeduren `int -> unit`, die für $n \in \mathbb{N}$ die folgenden Laufzeiten haben:

$\Theta(\log n)$	`fun tlogn n = ···`
$\Theta(n)$	`fun tn n = ···`
$\Theta(n \log n)$	`fun tnlogn n = ···`
$\Theta(n^2)$	`fun tn2 n = ···`

Für `tnlogn` und `tn2` dürfen Sie `tn` verwenden. Ansonsten sind keine Hilfsprozeduren erlaubt.